- **How do you structure your CSS and JavaScript to make it easier for other developers to work with?**

Front-end engineers quite often work with code developed by previous developers or cooperate with a team. So, this question allows you to find put whether developers are able to make websites or applications understandable for their colleagues. So, best answers should consist of:

- Examples that show experience in code organization
- Knowledge of what happens when the code isn't commented appropriately
- A desire to create easier things

Example: "I organize my stylesheets with sections for each site component. Each section has comments throughout the code so other developers can change it."

- **What is a closure in JavaScript?**

Example: A closure is referred to as an inner function having access to the variables in the outer function's scope chain. The closure has accesses to three scoped variables:

1. Variables in its own scope
2. Variables in the function's scope
3. Global variables

- **What is function hoisting? What are the two ways of creating a function?**

Example: In JavaScript, functions, and variables are hoisted. The JS interpreter looks forward to finding all the relevant declarations and then hoist them above the function, right where they are declared. There are two ways of creating functions in JavaScript.

1. Function Declaration defines the function with the specified parameters.
2. Function Expression defines a function as a part of a larger expression syntax.

```
// Function declaration
function add(num1, num2) {
    return num1 + num2;
}

// Function expression
var add = function (num1, num2) {
    return num1 + num2;
};
```

- **What is Event-driven programming?**

Example: Event-driven programming is building our application based on and respond to events. When an event occurs (e.g. click or keypress), a callback function is running to be registered to the element for that event.

- **Explain the differences between one-way data flow and two-way data binding**

Example: Two-way data binding means that UI fields are bound to model data dynamically, so when a UI field changes, the model data changes with it and vice-versa.
One way data flow means that the model is the single source of truth. As a result, only the model has access to change the app's state. The effect is that data always flows in a single direction, which makes it easier to understand.
One way data flows are deterministic, while two-way binding is able to cause side-effects which are harder to follow and understand.

- **What is the difference between Asynchronous and Non-blocking?**

Example: Asynchronous and Non-blocking look very similar in nature but there is a subtle difference here. Asynchronous means that our API will return immediately after calling it. In the background, it will start a process to fulfill the request. Once that process is complete, our work is done. But the main thread does not wait for that process to complete.
Non-blocking means if our API cannot complete the work, it returns an error immediately. Based on the response, success or error, we can decide if we want to make another call to the same API or continue with the next operation in our main flow. We can create a wait mechanism in a non-blocking operation.

- **How experienced are you with MEAN stack?**

Example: MEAN stack refers to the collection of technologies which are used to develop web applications. MEAN stands for "MongoDB Express.js AngularJS Node.js". The Node.js allows us to use JavaScript on the frontend as well as the backend. In a MEAN stack with NoSQL, nature of MongoDB you can build a product with clear attributes, and without worrying about migrations you can change and alter the data layers.
The stack is basically a set of languages and technologies utilized to create websites or applications by the developers.
Furthermore, it's good if a developer is able to provide you with any examples from his previous work or projects.

- **Explain the difference between Relational DB vs. NoSQL**

Databases are used for storing and managing large amounts of data. The relational model is useful when it comes to reliability but when it comes to the modern applications dealing with large amounts of data and the data is unstructured; non-relational models are usable. NoSQL databases are non-relational, distributed, open source and are horizontally scalable.
Each NoSQL data store addresses the various problems in relational databases. High availability, high scalability and fault tolerance provided by them. The different data stores use different techniques to achieve this goal and seem to suit well for their requirements.